บุญเลิศ เอี่ยมทัศนา

โปรแกรมด้วย Delphi

: หน้าต่างและไดอะล็อกบ็อกซ์

เปรียบเทียบกันในเรื่องของการสร้างและการใช้หน้าต่าง SDI, MDI และไดอะล็อกบ็อกซ์เมื่อโปรแกรมด้วยเดลไฟ ทั้งสามเรื่องนี้จะคล้ายกันมาก

เมื่อเขียนโปรแกรมกับวินโดวส์โดยตรง การสร้างหน้าต่าง SDI, MDI และไดอะล็อกบ็อกซ์ จะแยกจากกันตั้งแต่เริ่มต้น แต่เมื่อโปรแกรมด้วยเดลไฟ ทั้งสามเรื่องนี้จะคล้ายกันมาก

หน้าต่างกับไดอะล็อกบ็อกซ์

ข้อแตกต่างที่สำคัญของหน้าต่างธรรมดา (หน้าต่าง SDI) กับไดอะล็อกบ็อกซ์ คือ

1. การให้แสดงตัว กับหน้าต่างธรรมดาใช้ Show กับไดอะล็อกบ็อกซ์ใช้ ShowModal

2. ไดอะล็อกบ็อกซ์มีวงรอบการรันของตนเอง จึงมีผลตามที่จะกล่าวต่อไป สำหรับหน้าต่างธรรมดาใช้ของหน้าต่างหลัก

ดังจะแสดงตัวอย่างในโปรแกรม Wnd3Dlg (Window and Dialogbox) กล่าวคือ เมื่อรันโปรแกรมนี้จะแสดงหน้าต่างหลักดังรูปที่ 1 เมื่อเลือกปุ่ม Name จะแสดงหน้าต่างรองดังรูปที่ 2 แต่ถ้าเลือกปุ่ม Address จะแสดงไดอะล็อกบ็อกซ์มีรูปแบบเช่นเดียวกับหน้าต่างรองในรูปที่ 2 ซึ่งมีรายละเอียดอื่นอีกตามที่จะกล่าวต่อไป

เตรียมโปรแกรม

เตรียมโปรแกรมของหน้าต่างหลักตามรูปที่ 1 และตามลิสติงที่ 1 ของ Form2 ตามรูปที่ 2 และลิสติงที่ 2 ของ Form2 ซึ่งจะใช้เป็นทั้งหน้าต่างธรรมดาและไดอะล็อกบ็อกซ์ มีรายละเอียดเป็นพิเศษดังนี้ คือ

1. จะต้องเปิดฟอร์มเพิ่มโดยเลือกรายการ File/New Form ซึ่งจะได้ทั้งฟอร์มและยูนิต

2. ให้ค่า BorderStyle เป็น bsDialog จะทำให้เป็นหน้าต่างที่มีรูปแบบของไดอะล็อกบ็อกซ์คือ

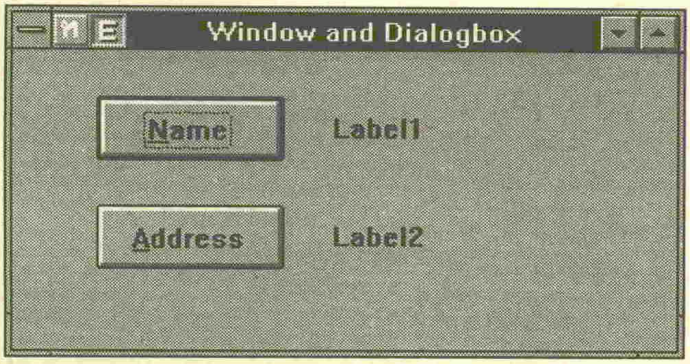
- ไม่มีปุ่ม Minimize และ Maximize
- มีกรอบหน้าต่างที่ย่อขยายไม่ได้
- ที่ System เมนู มีรายการเพียง 2 รายการคือ Move กับ Close

3. ให้ค่าพรอพเพอร์ตี้ส์ของปุ่ม OK ตามแบบของไดอะล็อกบ็อกซ์คือ

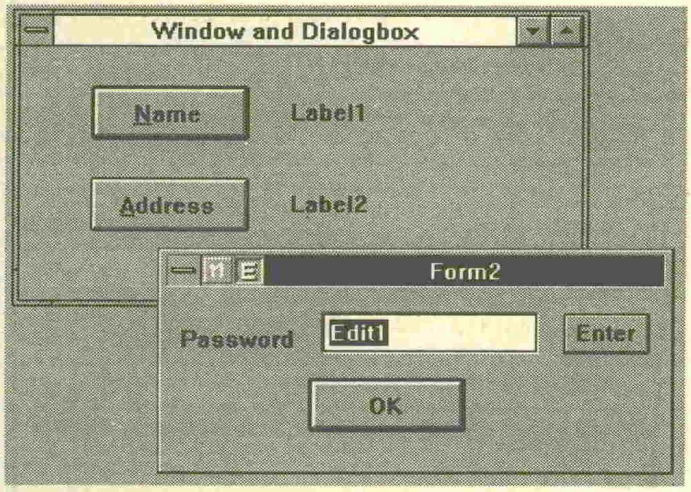
- ให้ค่า Default เป็น True (ถ้าเป็นปุ่ม Cancel ให้ค่า Cancel เป็น True)
- ให้ค่า ModalResult เป็น mrOK (ถ้าเป็นปุ่ม Cancel จะต้องให้เป็น mrCancel)

คำอธิบาย

1. ในโปรแกรมของกิจกรรม Button1Click ของหน้าต่างหลัก (ลิสติงที่ 1) ซึ่งตอบสนองการกดปุ่ม Name ได้ให้แสดง Form2 ด้วย Show ทำให้ได้เป็นหน้าต่างธรรมดา นั่นคือ



รูปที่ 1 หน้าต่างหลักของโปรแกรม Wnd3Dlg



รูปที่ 2 แสดงหน้าต่าง Form2 ซึ่งจะใช้เป็นทั้งหน้าต่างและไดอะล็อกบ็อกซ์

- ผู้ใช้สามารถกลับไปทำงานในหน้าต่างหลักได้ เช่น เลือกปุ่ม Address (แต่ถ้าเลือกจะเกิด run-time error เพราะจะเป็นการใช้ฟอร์มเดียวกันเป็นทั้งหน้าต่างและไดอะล็อกบ็อกซ์ ในขณะเดียวกัน)

- ได้เตรียมการไว้ว่า เมื่อผู้ใช้ป้อนรหัสผ่านแล้วให้เลือกปุ่ม Enter (เพื่อให้แตกต่างจากปุ่ม OK เท่านั้น)

- เมื่อผู้ใช้เลือกปุ่ม Enter จะมาทำงานที่กิจกรรม Speed Button1Click ซึ่งได้ให้ค่าที่ผู้ใช้ป้อนแก่กรอบ Label1 ของหน้าต่างหลัก ทั้งนี้เพราะเป็นการยากที่จะเขียนโปรแกรมให้ดำเนินการเช่นนี้ในหน้าต่างหลัก เสร็จแล้วให้ปิดหน้าต่าง Form2 ด้วย Close

- แต่การปิดหน้าต่างจริงอยู่ในกิจกรรม FormClose ซึ่งได้ให้ตรวจรหัสผ่านที่ผู้ใช้ป้อนว่าถูกต้องหรือไม่ ถ้าถูกต้องจึงให้หายตัวด้วย caHide (เพื่อให้หายตัวไม่ใช่ปิด ตามที่จะกล่าวต่อไป

2. เมื่อผู้ใช้เลือกปุ่ม Address จะมาทำงานที่กิจกรรม Button2 Click ซึ่งให้แสดง Form2 ด้วย ShowModal เท่ากับให้เป็นไดอะล็อก-บ็อกซ์ จะเกิดผลดังนี้

ลิสติงที่ 1

```
PROGRAM  Wnd3Dlg;  { wnd3dlg.dpr }

Form == Form1 ( Window and Dialogbox }
Unit == Wnd3Dlg1  { wnd3dlg1.pas }

Components:-
1 Standard :: Button == Button1 ( Name )
2 Standard :: Label == Label1
3 Standard :: Button == Button2 ( Address )
4 Standard :: Label == Label2


(*-------- Implementation part ---------*)

procedure TForm1.Button1Click(Sender: TObject);
begin
  Form2.Show;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  Form2.ShowModal;
  if  Form2.ModalResult = mrOK  then
    Label2.Caption := Form2.Edit1.Text
end;

end.
```

ลิสติงที่ 2

```
{ program Wnd3Dlg }

Form == Form2
        - BorderStyle = bsDialog
Unit == Wnd3Dlg1  { wnd3dlg1.pas }

Components:-
1 Standard :: Label == Label1 ( Password )
2 Standard :: Edit == Edit1
3 Additional :: SpeedButton == SpeedButton1
( Enter )
4 Standard :: Button == Button1 ( OK )

(*-------- Implementation part ---------*)

uses  Wnd3Dlg1;

procedure TForm2.FormClose(Sender: TObject;
var Action: TCloseAction);
begin
   Action := caHide;
   if  ModalResult = mrCancel  then  Exit;
   if  Edit1.Text <> 'aeiou'  then
         Action := caNone;
end;


procedure TForm2.SpeedButton1Click(Sender:
TObject);
begin
  Form1.Label1.Caption := Edit1.Text;
  Close;
end;


end.
```

- โดยปกติเมื่อให้แสดงหน้าต่าง (ไดอะล็อกบ็อกซ์เป็นหน้าต่างเช่นกัน) หน้าต่างนั้นจะได้รับโฟกัส

- แต่โดยที่ในขณะที่ Form2 เป็นไดอะล็อกบ็อกซ์ มีวงรอบการรันของตนเอง จึงไปทำงานที่วงรอบการรันนี้ ไม่ปฏิบัติประโยค if ที่ตามมา

- และผู้ใช้ไม่สามารถกลับมาใช้หน้าต่างหลัก เช่น เมื่อเลือกปุ่ม Name จะไม่เกิดผลใด

- เมื่อผู้ใช้ป้อนรหัสผ่านแล้วกดคีย์ Enter จะเทียบเท่ากับเลือกปุ่ม OK เพราะได้ให้ค่า Default ไว้เป็น True

- เมื่อผู้ใช้เลือกปุ่ม OK จะปิดหน้าต่างโดยอัตโนมัติ ให้ค่า ModalResult ตามที่ให้ไว้แก่ปุ่มนี้คือ mrOK แต่ถ้าเลือกรายการ Close จะได้ค่าที่เป็น mrCancel และปิดหน้าต่างเช่นกัน

- ซึ่งการปิดหน้าต่างจริงจะอยู่ในกิจกรรม FormClose ตามที่กล่าวมา และความจริงแล้วการเขียนโปรแกรมของกิจกรรมนี้เป็นวิธีการเขียนโปรแกรมให้แก่ไดอะล็อกบ็อกซ์ ซึ่งของหน้าต่างไม่จำเป็นต้องมีกิจกรรม FormClose คงให้รวมอยู่ใน SpeedButton1Click ได้

จะเห็นว่าสามารถใช้หน้าต่างให้เป็นได้ทั้งหน้าต่างธรรมดา และไดอะล็อกบ็อกซ์

หน้าต่าง SDI กับ MDI

ผมเข้าใจว่าคำว่า SDI (Single Document Interface) ตั้งขึ้นเพื่อให้เข้าคู่กับ MDI (Multi Document Interface) โดยที่ชุดหน้าต่าง SDI คือหน้าต่างธรรมดาโดยทั่วไป อาจประกอบด้วยหน้าต่างหลักเพียงบานเดียว หรืออาจมีหน้าต่างรองบานหนึ่งหรือหลายบาน ตามความต้องการใช้งาน ดังเช่นหน้าต่างของเคลไฟ

ส่วนชุดหน้าต่าง MDI จะประกอบด้วยหน้าต่างแม่บานหนึ่งเรียกเป็น Frame window และมีหน้าต่างลูกเรียกเป็น Child window ชุดหนึ่งหรือหลายชุด ซึ่งผู้ใช้จะเปิดใช้หน้าต่างลูกแต่ละชุดได้หลายบานตามต้องการ เช่น หน้าต่างของโปรแกรม Database Desktop

และความจริงที่พื้นที่ใช้งานของหน้าต่างแม่ ยังมีหน้าต่างอีก 1 บาน เรียกว่า Client Window ทำหน้าที่เกี่ยวกับหน้าต่างลูกเช่นการ Tile และ Cascade เป็นต้น แต่เมื่อโปรแกรมด้วยเคลไฟจะกลืนอยู่กับหน้าต่างแม่ ไม่อ้างถึง

เพื่อให้ได้เปรียบเทียบกันทั้งในเรื่องของการสร้างและการใช้ จะเขียนโปรแกรม 2 โปรแกรมให้ทำงานเหมือนกันคือ SDIWnd เป็นหน้าต่างในชุด SDI และ MDIWnd เป็นหน้าต่างในชุด MDI

โปรแกรม SDIWnd

โปรแกรมนี้จะแสดงการใช้หน้าต่างในชุด SDI โดยที่เมื่อรันจะแสดงหน้าต่างหลัก ซึ่งถ้าผู้ใช้เลือกรายการ

ลิสติงที่ 3

```
PROGRAM  SDIWnd;  { sdiwnd.dpr }

Form == Form1 { SDI Window }
Unit == Main1 { main1.pas }
        - unit = + UTxtWnd2,UBmpWnd3

Components:-
1 Standard :: MainMenu == MainMenu1
  1.0 Main = ( &File, &Window )
  1.1 File = ( E&xit )
      - GroupIndex = 0
  1.2 Window = ( Open &Text, Open &Picture )
      - GroupIndex = 9


(*-------- Implementation part ---------*)

procedure TForm1.FormCreate(Sender: TObject);
begin
  TxtForm2 := nil;  BmpForm3 := nil;
end;

procedure TForm1.Exit1Click(Sender: TObject);
begin
  Close;
end;

procedure TForm1.OpenText1Click(Sender:
TObject);
begin
  if  TxtForm2 = nil  then
     Application.CreateForm(TTxtForm2,
           TxtForm2);
  TxtForm2.Show;
end;

procedure TForm1.OpenPicture1Click(Sender:
TObject);
begin
  if  BmpForm3 = nil  then
    Application.CreateForm(TBmpForm3,
          BmpForm3);
  BmpForm3.Show;
Label1.Caption := IntToStr(Screen.FormCount);
end;

end.
```

- Window/Open Text จะแสดงหน้าต่าง Text Window คือ TxtForm2 โดยสมมติว่าเป็นหน้าต่างประเภทเดียวกับ Notepad

- Window/Open Picture จะแสดงหน้าต่าง Bitmap Window คือ BmpForm3 โดยสมมติว่าเป็นหน้าต่างประเภทเดียวกับ Paintbrush

จะเตรียมโปรแกรมได้ดังนี้

- ของหน้าต่างหลักตามลิสติงที่ 3

- ของหน้าต่าง TxtForm2 ตามลิสติงที่ 4 โดยเปิดฟอร์มใหม่

- ของหน้าต่าง BmpWnd3 ตามลิสติงที่ 5 โดยเปิดฟอร์มใหม่

เมื่อเตรียมโปรแกรมแล้วยังรันไม่ได้ เพราะผมได้เพิ่มเรื่องของหน้าต่างปิดตามที่จะกล่าวต่อไป

ลิสติงที่ 4

```
{ program  SDIWnd }

Form == TxtForm2 ( Text Window )
Unit == Utxtwnd2 { utxtwnd2.pas }
        - uses = + ClipBrd

Components :-
1 Standard :: MainMenu == MainMenu1
  1.0 Main = ( &File, &Edit, &Font )
  1.1 File = ( &Open, Save &As, -, E&xit )
      - GroupIndex = 0
  1.2 Edit = ( Cu&t, &Copy, &Paste )
      - GroupIndex = 1
  1.3 Font.GroupIndex = 1
2 Standard :: Memo == Memo1
  - Align = alClient
3 Dialogs :: OpenDialog == OpenDialog1
  - DefaultExt = txt
  - FileName = *.txt
  - Filter = [ Text File (*.txt), All Files
(*.*) ]
4 Dialogs :: SaveDialog == SaveDialog1
  - DefaultExt = txt
  - FileName = *.txt
  - Filter = [ Text File (*.txt), All Files
(*.*) ]

(*------- Implementation part ---------*)

{$R *.DFM}

procedure TTxtForm2.Exit1Click(Sender:
TObject);
begin
  Close;
end;

procedure TTxtForm2.Open1Click(Sender:
TObject);
begin
  if  OpenDialog1.Execute  then
    Memo1.LineS.LoadFromFile(OpenDialog1.FileName);
end;

procedure TTxtForm2.SaveAs1Click(Sender:
TObject);
begin
  if  SaveDialog1.Execute  then
   Memo1.LineS.SaveToFile(OpenDialog1.FileName);
end;

procedure TTxtForm2.Edit1Click(Sender:
TObject);
var  selDta : Boolean;
begin
  selDta := Memo1.SelLength > 0;
  Cut1.Enabled := selDta;
  Copy1.Enabled := selDta;
  if  Clipboard.HasFormat(cf_Text)  then
    Paste1.Enabled := True
  else  Paste1.Enabled := False;
end;

procedure TTxtForm2.Cut1Click(Sender:
TObject);
begin
   Memo1.CutToClipboard;
end;

procedure TTxtForm2.Copy1Click(Sender:
TObject);
begin
   Memo1.CopyToClipboard;
end;

procedure TTxtForm2.Paste1Click(Sender:
TObject);
begin
  Memo1.PasteFromClipboard;
end;


end.
```

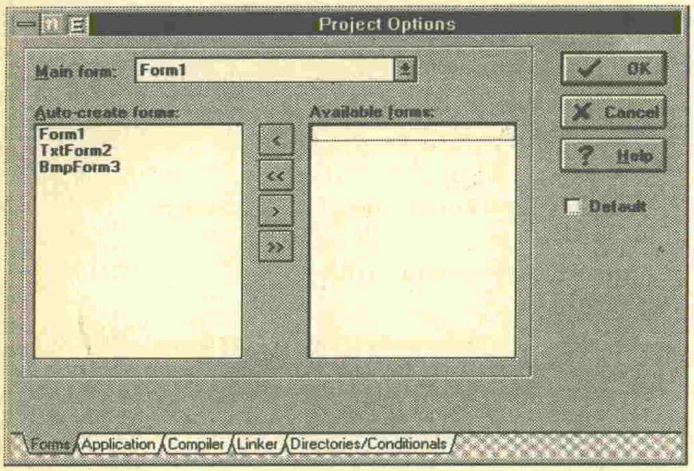
ลิสติงที่ 5

```
{ program  SDIWnd }

Form == BmpForm3 ( Bitmap Window )
Unit == UBmpwnd3 { ubmpwnd3.pas }
        - uses = + ClipBrd

Components :-
1 Standard :: MainMenu == MainMenu1
  1.0 Main = ( &File, &Edit)
  1.1 File = ( &Open, Save &As, -, E&xit )
      - GroupIndex = 0
  1.2 Edit = ( &Get Clipboard, &Put Clipboard )
      - GroupIndex = 1
2 Additional :: Image == Image1
  - AutoSize = True
3 Dialogs :: OpenDialog == OpenDialog1
  - DefaultExt = bmp
  - FileName = *.bmp
  - Filter = [ Bmp File (*.bmp), All Files
(*.*) ]
4 Dialogs :: SaveDialog == SaveDialog1
  - DefaultExt = bmp
  - FileName = *.bmp
  - Filter = [ Bmp File (*.bmp), All Files
(*.*) ]

(*------- Implementation part ---------*)

{$R *.DFM}

procedure TBmpForm3.Exit1Click(Sender:
TObject);
begin
  BmpForm3 := nil;  Release;
end;

procedure TBmpForm3.Open1Click(Sender:
TObject);
begin
  if  OpenDialog1.Execute  then
   Image1.Picture.LoadFromFile(OpenDialog1.FileName);
end;

procedure TBmpForm3.SaveAs1Click(Sender:
TObject);
begin
```

⇨

ลิสติงที่ 5 (ต่อ)

```
  if  SaveDialog1.Execute  then
   Image1.Picture.SaveToFile(SaveDialog1.FileName);
end;

procedure TBmpForm3.Edit1Click(Sender:
TObject);
begin
  if  Clipboard.HasFormat(cf_Bitmap)  then
    Get1.Enabled := True
  else  Get1.Enabled := False;

  if  Image1.Picture.Bitmap.Empty  then
    Put1.Enabled := True
  else  Put1.Enabled := False
end;

procedure TBmpForm3.Get1Click(Sender: TObject);
begin
  Image1.Picture.Assign(Clipboard);
end;

procedure TBmpForm3.Put1Click(Sender: TObject);
begin
  Clipboard.Assign(Image1.Picture)
end;


end.
```



รูปที่ 3 แสดงหน้า Forms ของไดอะล็อกบ็อกซ์ Project Options มีรายการฟอร์ม เมื่อเริ่มแรกอยู่ทางด้านซ้ายเป็นหน้าต่างเปิด เพื่อให้เลือกไปไว้ทางด้านขวาเป็นหน้าต่างปิด

หน้าต่างเปิดและหน้าต่างปิด

เมื่อเตรียมโปรแกรมตามที่กล่าวมานี้ จะได้หน้าต่างรองเป็นหน้าต่างเปิดคือ โปรแกรมที่เคลไฟเตรียมไว้ให้เรา จะให้กำเนิดหน้าต่างเหล่านี้ตั้งแต่เริ่มต้น ดังนั้นเมื่อมีหน้าต่างรองมาก นอกจากจะเปลืองหน่วยความจำแล้ว ยังทำให้หน้าต่างหลักแสดงตัวช้าอีกด้วย

ควรกำหนดให้หน้าต่างที่ไม่ได้ใช้งานแต่เริ่มต้นเป็นหน้าต่างปิด และมีเรื่องที่เกี่ยวพันกันดังนี้

1. ทำเป็นหน้าต่างปิดได้ โดย

- เลือกรายการ Options/Project จะแสดงไดอะล็อกบ็อกซ์ ให้เลือกหน้า Forms ดังในรูปที่ 3

- ในกรอบรายการทางด้านซ้ายจะแสดงรายชื่อหน้าต่าง ในกรณีนี้ต้องการให้ทั้งหน้าต่าง TxtForm2 และ BmpForm3 เป็นหน้าต่างปิด ให้เลือกหน้าต่างทั้งสองนี้ แล้วเลือกปุ่มหัวลูกศรขวา แล้วเลือกปุ่ม OK

2. เมื่อจะใช้งานหน้าต่างปิด จะต้องให้กำเนิดก่อนด้วย Create Form และให้ Show ด้วย ดังที่เห็นในโปรแกรมของกิจกรรม OpenText1Click และ OpenPicture1Click ในลิสติงที่ 3

3. แต่โดยที่เมื่อผู้ใช้ปิดหน้าต่างรอง เช่น โดยเลือกรายการ Close หรือเราให้ปิดด้วยคำสั่ง Close จะเป็นแค่เพียงการหายตัวเท่านั้น ดังนั้นถ้าผู้ใช้เปิดหน้าต่างเหล่านี้ใหม่ และเราให้กำเนิดใหม่จะเป็นหน้าต่างบานใหม่ ส่วนบานเดิมจะตกค้างอยู่ในหน่วยความจำไม่ได้นำมาใช้ จึงได้ให้ตรวจก่อนให้กำเนิด

4. ในกรณีที่เป็นหน้าต่างที่ใช้น้อยไม่ต้องการให้เป็นหน้าต่างเปิดอยู่ตลอดเวลา จะต้องให้ปิดด้วยการยุบด้วยคำสั่ง Release ดังที่แสดงไว้ในโปรแกรมของกิจกรรม Exit1Click ของ BmpForm3 ในลิสติงที่ 5

สำหรับโปรแกรมนอกนั้นเป็นเรื่องที่เคยกล่าวมาแล้ว จึงของดคำอธิบายครับ

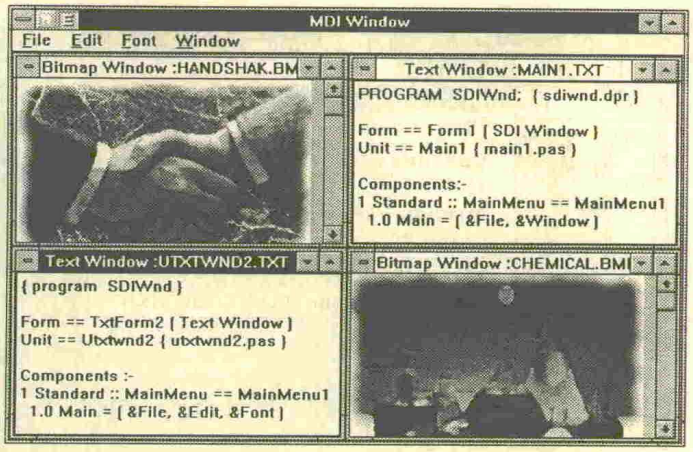
ทดลองเปิดหน้าต่างซ้ำ

หน้าต่างรองทั้งสองบานตามที่กล่าวมา กำเนิดมาจากสกุล TTxtForm2 และ TBmpForm3 ตามลำดับ สกุลซึ่งต่อไปในเรื่องของหน้าต่าง MDI จะเรียกเป็นชุด สกุลละ 1 บาน ซึ่งเป็นวิธีการตามปกติของการใช้หน้าต่าง SDI

โดยที่ในวินโดวส์ได้วางแนวไว้ว่า หากต้องการใช้หน้าต่างของสกุลเดียวกันหลายบาน ให้ใช้เป็นหน้าต่าง MDI แต่ทดลองดูได้ เช่น ถ้าต้องการให้เปิดใช้หน้าต่าง Text Window ได้มากกว่า 1 บาน ให้ดำเนินการดังนี้

1. ในโปรแกรมของกิจกรรม OpenText1Click ให้ทำหมายเหตุคร่อม if ถึง then

2. รันโปรแกรมนี้ แล้วเลือกรายการ Window/Open Text จะได้หน้าต่าง Text Window หนึ่งบาน



รูปที่ 4 ชุดหน้าต่าง MDI มีหน้าต่างลูก 4 บาน โดยที่บานทางด้านล่างซ้ายแอคทีฟ

3. เลือกรายการนี้ซ้ำ จะได้หน้าต่างใหม่อีก 1 บาน (ซ้อนทับหน้าต่างเดิม)

โปรแกรม MDIWnd

หน้าต่างในชุด MDI ประกอบด้วยหน้าต่างแม่ (หน้าต่างหลัก) 1 บาน กับหน้าต่างลูกอีกหนึ่งหรือหลายชุด โดยที่ผู้ใช้สามารถเปิดหน้าต่างลูกในแต่ละชุดได้หลายบาน ดังเช่นในรูปที่ 4 ได้เปิดหน้าต่างลูกชุดละ 2 บาน และให้จัดเรียงในแบบ Tile

สามารถจะเตรียมโปรแกรมให้แก่หน้าต่างในชุด MDI ได้เช่นเดียวกับหน้าต่างในชุด SDI แต่มีข้อแตกต่างกันและจะขออ้างถึงโปรแกรม SDIWnd เพื่อการเปรียบเทียบ โดยที่จะแปลงโปรแกรม SDIWnd เป็นโปรแกรม MDIWnd ได้ดังนี้

1. ชนิดของหน้าต่าง

1.1 จะต้องให้ค่า FormStyle ดังนี้

- ของหน้าต่างแม่ให้เป็น fsMDIForm
- ของหน้าต่างลูก (คือ TextForm2 และ BmpForm3) ให้เป็น fsMDIChild

1.2 หน้าต่างลูกจะต้องเป็นหน้าต่างปิด (เป็นอยู่แล้ว)

2. การให้กำเนิดและยุบหน้าต่าง

2.1 ให้กำเนิดด้วย Create (เช่นเดียวกับให้กำเนิดคอนโทรล) และไม่ต้องให้ Show ดังโปรแกรมของหน้าต่างแม่ในลิสติงที่ 6

2.2 ผู้ใช้จะปิดหน้าต่างลูกแต่ละบานได้โดยเลือกรายการ Close (ใน System Memu) ซึ่งเราจะต้องจัดกิจกรรมสนองตอบอีเวนต์ OnClose เช่น

ลิสติงที่ 6

```
PROGRAM  MDIWnd; { mdiwnd.dpr }

Form == Form1 ( MDI Window }
Unit == Main1 { main1.pas }
        - unit = + UTxtWnd2,UBmpWnd3

Components:-
1 Standard :: MainMenu == MainMenu1
  1.0 Main = ( &File, &Window )
  1.1 File = ( E&xit )
      - GroupIndex = 0
  1.2 Window =( Open &Text,Open &Picture,&Cascade,
                &Tile, &Arrange Icons,Close &All )
      - GroupIndex = 9

(* ——— Implementation part ———— *)

procedure TForm1.FormCreate(Sender:TObject);
begin
  TxtForm2 := nil;  BmpForm3 := nil;
end;

procedure TForm1.Exit1Click(Sender:TObject);
begin
  Close;
end;

procedure TForm1.OpenText1Click(Sender:TObject);
begin
  TTxtForm2.Create(Application);
end;

procedure TForm1.OpenPicture1Click(Sender:TObject);
begin
  TBmpForm3.Create(Application);
end;

procedure TForm1.Cascade1Click(Sender:TObject);
begin
  Cascade;
end;

procedure TForm1.Tile1Click(Sender:TObject);
begin
  Tile
end;

procedure TForm1.ArrangeIcons1Click(Sender:TObject);
begin
  ArrangeIcons
end;

procedure TForm1.CloseAll1Click(Sender:TObject);
var  chd : Integer;
begin
  for chd := MDIChildCount - 1 downto 0 do
    MDIChildren[chd].Close
end;

end.
```

```
TTxtForm2.FormClose
```

เพื่อตรวจสอบความพร้อม เช่น ตรวจว่าหากผู้ใช้ป้อนแก้ได้เก็บเอกสารลงไฟล์แจ้งหรือยัง หากถูกต้องจะต้องให้ค่า caFree แก่ Action เพื่อให้ปิดหน้าต่างบานนั้นด้วยการยุบ

3. การจัดระเบียบหน้าต่างลูก

3.1 การจัดระเบียบ ควรจัดให้มีรายการ Cascade, Tile, Arrange Icons และ Close All ดังแสดงในรูปที่ 5 (ขอให้เพิ่มรายการเหล่านี้ให้แก่ MainMemu1 ของหน้าต่างแม่ และเตรียมโปรแกรมดังในลิสติงที่ 6)

3.2 เป็นวิธีการของวินโดวส์อย่างหนึ่งที่จะแสดงรายชื่อหน้าต่างลูก (ชื่อที่ไตเติลบาร์) ในรายการ Window ดังที่เห็นในรูปที่ 5 ซึ่งเราจะให้อยู่ในรายการใดก็ได้เช่น File หรือ Window โดยให้ค่า (พรอพเพอร์ตี้ส์) แก่ WindowMemu ของหน้าต่างแม่ (ขอให้ให้ค่าโดยเลือกเป็น Window1)

ขอให้สังเกตด้วยว่า ตามรูปที่ 5 เป็นภาพของหน้าต่างหลักในขณะเตรียมโปรแกรม ซึ่งที่เมนูมีรายการหลักเพียง 2 รายการคือ File กับ Window

4. การสนธิเมนู

เมื่อเริ่มต้นรันโปรแกรมนี้จะแสดงเมนูของหน้าต่างแม่ต่อเมื่อเปิดหน้าต่างลูก และเมื่อเปิดหน้าต่างลูกแล้วหลายบาน หากผู้ใช้เลือกหน้าต่างบานใดจะแสดงเมนูของหน้าต่างลูกชุดนั้น แต่เป็นเมนูที่ได้สนธิ (merge) เมนูของหน้าต่างแม่และของหน้าต่างลูกแล้ว ดังที่เห็นในรูปที่ 4 และ 5 มีวิธีการสนธิเมนูและเรื่องที่เกี่ยวข้องดังนี้

4.1 จะต้องให้ค่า ซึ่งจะขอเรียกเป็นค่าลำดับที่ แก่ GroupIndex ของรายการหลักทุกรายการในทุกเมนู โดยที่

- ค่าที่ให้ได้คือ 0 ถึง 255
- อาจให้ค่าลำดับที่แก่แต่ละรายการ หรืออาจให้ค่าลำดับที่เดียวกันแก่หลายรายการ

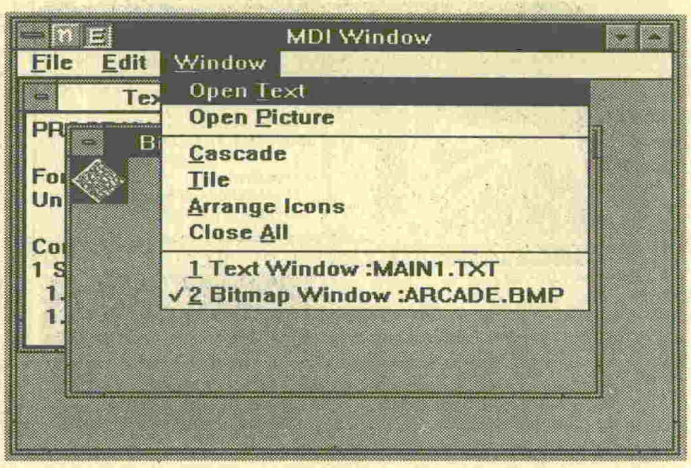
4.2 ในการสนธิ จะรวมรายการหลักของทั้งสองเมนู (ของหน้าต่างแม่กับหน้าต่างลูก) เข้าด้วยกัน โดยจัดเรียงลำดับตามค่าลำดับที่ แต่ถ้าค่าลำดับที่ซ้ำกัน จะใช้รายของหน้าต่างลูก ซึ่งส่วนมากจะได้แก่ File

ตามตัวอย่างนี้คือในลิสติงที่ 3, 4 และ 5 ได้เตรียมการไว้แล้ว โดยให้ค่าลำดับที่แก่ GroupIndex ของรายการต่าง ๆ ของเมนูหลักของฟอร์มต่าง ๆ ซึ่งมีชื่อเหมือนกันหมดคือ MainMemu1 ไว้ดังนี้คือ

- ของ Form1 คือ File=0 และ Window=9
- ของ TxtForm2 คือ File=0 และ Edit กับ Font=1
- ของ BmpForm3 คือ File=0 และ Edit=1

5. โปรแกรมของหน้าต่างแม่

ตามที่กล่าวมานี้ เมื่อมีรายการซ้ำกันจะใช้ของหน้าต่างลูกซึ่งหมายถึงโปรแกรมด้วย ดังนั้นในโปรแกรมของหน้าต่างลูกจะต้องดำเนินการดังนี้



รูปที่ 5 แสดงรายการใน Window เมื่อเปิดหน้าต่างลูก 2 บาน และบานบนแอคทีฟ

ลิสติงที่ 7

```
implementation

{$R *.DFM}

uses  Main1;

procedure TTxtForm2.Exit1Click(Sender: TObject);
begin
  Form1.Exit1Click(Sender);
end;
```

5.1 ในรายการหลักที่ซ้ำกัน กรณีนี้คือ File นอกจากรายการรองของตนแล้ว จะต้องมีรายการรองของหน้าต่างแม่ด้วย คือ Exit (แต่บังเอิญผมเตรียมการไว้แล้วจึงดูเหมือนไม่เพิ่ม)

5.2 จะต้องจัดกิจกรรมสนองตอบรายการที่เพิ่มมานี้ เช่นเดียวกับที่เป็นของตน แต่ทำงานของหน้าต่างแม่ ซึ่งส่วนใหญ่จะใช้วิธีเรียกใช้กิจกรรมของหน้าต่างแม่ ดังแสดงในกิจกรรม Exit1Click ในลิสติงที่ 7 โดยที่จะต้องบ่งใช้ยูนิตของหน้าต่างแม่คือ Main1 ใน uses หลังคำว่า Implementation ด้วย

ท้ายตอน

หวังว่าในฉบับนี้คงจะพอทำให้เห็นว่า เมื่อใดควรเป็นหน้าต่าง SDI เมื่อใดควรเป็น MDI และเมื่อใดควรทำเป็นไดอะล็อกบ็อกซ์ และจากสองฉบับก่อนเพื่อตอบปัญหาว่าทำไมถึงต้องมีหน้าต่างหลายบานครับ